CLAIMS

1. A method for updating a software component in a distributed software system comprising a first software component and one or more second software components, said method comprising

updating said first software component with a new version of software substantially compatible with the functionality of a non-updated version(s) of said one or more second software components but having data structures incompatible with said non-updated version(s) of said one or more second software components,

providing said updated software version with a data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange,

delivering said metadata description with said new version of said software component.

2. A method as claimed in claim 1, further comprising

providing a dedicated data exchange metadata description for each version of said second software component that should be compatible, each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

delivering each of said dedicated data exchange metadata descriptions with the delivery of the respective version of said second software component,

delivering said dedicated data exchange metadata description separately for each data exchange session during a session handshake procedure between said new version of said software component and the respective version of said second software component.

3. A method as claimed in claim 1 or 2, further comprising

verifying the backward compatibility of said new version of said first software component prior to said delivery on the basis of said metadata description of said new version and the metadata description of previous version(s) of said second software component.

4. A method as claimed in claim 1 or 2, comprising

delivering said first software component to a server computer to be used as a server software component in a client-server-type distributed software system, said one or more second software components being client software components located in respective client stations.

5. A method for updating a software component in a distributed software system comprising a first software component and one or more second software components, said method comprising

updating said first software component with a new version of software substantially compatible with the functionality of a non-updated version(s) of said one or more second software components but having data structures incompatible with said non-updated version(s) of said one or more second software components,

providing said updated software version with a data exchange metadata description containing information on data structures to be used in a serialized data exchange,

providing a dedicated data exchange metadata description for each version of said second software component that should be compatible, each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

delivering said data exchange metadata descriptions with said new version of said software component.

6. A method as claimed in claim 5, further comprising

verifying the backward compatibility of said new version of said first software component prior to said delivery by comparing said metadata description of said new version with the metadata description of previous version(s) of said second software component.

7. A method for updating a software component in a distributed software system comprising a first software component and one or more second software components installed apart from each other, said method comprising

updating said first software component with a new version of software substantially compatible with the functionality of a non-updated version(s) of said one or more second software components but having data structures incompatible with said non-updated version(s) of said one or more second software components,

providing said updated software version with a data exchange metadata description containing information on data structures to be used in a serialized data exchange,

providing a dedicated data exchange metadata description for each version of said second software component that should be compatible, each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

verifying backward compatibility for at least one pair of said new version of said first software component and a previous version of said second software component on the basis of said metadata descriptions of these versions.

8. A method for updating a software component in a distributed software system comprising a first software component and one or more second software components installed apart from each other, said method comprising

updating said first software component with a new version of software substantially compatible with the functionality of a non-updated version(s) of said one or more second software components but having data structures incompatible with said non-updated version(s) of said one or more second software components,

providing said updated software version with a data exchange metadata description containing information on data structures to be used in a serialized data exchange,

providing a dedicated data exchange metadata description for each version of said second software component that should be compatible, each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

verifying backward compatibility and creating a serialization scheme for at least one pair of said new version of said first software component and a previous version of said second software component on the basis of said metadata descriptions of these versions, said serialization scheme containing information needed in said new version of said first software component for the serialization of data to and the deserialization of data from said respective previous version of said second software component,

delivering said at least one serialization scheme with said new version of said software component.

9. A method as claimed in claim 8, wherein said verifying comprises

loading the data exchange metadata descriptions of said version and a previous version of said first software component,

comparing the identifier and type information of data structures in said new version and said previous version,

checking that all compatible structures exist,

checking that data items with compatible types exist, and

checking that there is a default value for each data item added to or removed from said new version in comparison with said previous version.

10. A method for exchanging data between software components in a distributed software system comprising a first software component and a second software component, said method comprising

providing said first software component with a first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said first software component,

providing said first software component with a dedicated second data exchange metadata description for any older version of said second software component that should be compatible, each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

said first software component identifying a version of a second software component,

said first software component selecting one of said second data exchange metadata descriptions corresponding to said identified version of said second software component,

said first software component serializing data items using data structures according to said first data exchange metadata description and said selected second data exchange metadata description, and sending said serialized data items to said second software component,

said first software component receiving serialized data items from said second software component and deserializing said data items using data structures according to said first data exchange metadata description and said selected second data exchange metadata description.

11. A method for exchanging data between software components in a distributed software system comprising a server software component and client software components, said method comprising

providing said server software component with a first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said server software component,

said server software component identifying a version of a client software component,

said server software component selecting according to said identified version of said client component a second data exchange metadata description, said second data exchange metadata description containing information on data structures for said identified version of said client software component that are incompatible with the data structures of the version of said server software component,

said server software component serializing data items on the basis of said first data exchange metadata description and said selected second data exchange metadata description, and sending said serialized data items to said client software component,

said server software component receiving serialized data items from said client software component and deserializing said data items on the basis of said first data exchange metadata description and said selected second data exchange metadata description.

12. A method as claimed in claim 11, further comprising delivering said selected second data exchange metadata description either by

delivering said second metadata description with said server software component and storing it said server software component, or by

delivering said second metadata description during a data exchange session performed between said server software component and said identified version of said client software component.

13. A method as claimed in claim 11, comprising

creating a serialization scheme for said server software component and said identified version of said client software component on the basis of said metadata descriptions of these versions, said serialization scheme containing information needed in said server software component for the serialization of data to and the deserialization of data from said version of said client software component.

14. A method as claimed in claim 13, comprising

storing the created serialization scheme in said server software component,

using said stored serialization scheme in a subsequent data exchange session between said server software component and said identified version of said client software component.

15. A method as claimed in claim 11, wherein

said step of serializing comprises transforming data items into a stream of data bits to be sent to said client software component,

said step of deserializing comprises transforming a stream of bits received from said client software components into data items.

16. A method as claimed in claim 15, comprising

assigning a short identifier for each distinct instance of a data structure transferred in a data exchange operation,

sending and serializing that short identifier for each occurrence of each instance of a data structure,

serialising and sending each actual data structure instance only for the first occurrence within the data exchange operation instead of serializing

the actual data structure instance for each occurrence within the data exchange operation.

17. A method for exchanging data between software components in a distributed software system comprising a first software component and a second software component, said method comprising

storing in said first software component a serialization scheme for at least one pair of said new version of said first software component and an older version of said second software component, said serialization scheme containing information needed in said new version of said first software component for the serialization of data to and the deserialization of data from said respective previous version of said second software component, and said serialization scheme being created on the basis of a first data exchange metadata description and a second data exchange metadata description, said first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said first software component, and said second data exchange metadata description of said older version of said second software component containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

said first software component identifying a version of a second software component,

said first software component selecting serialization scheme corresponding to said identified version of said second software component,

said first software component serializing data items on the basis of said selected serialization scheme,

said first software component receiving serialized data items from said second software component and deserializing said data items on the basis of said selected serialization scheme.

18. A distributed computer system, comprising

a first software component,

one or more second software components exchanging serialized data with said first software component,

at least one of said second software components being an older version substantially compatible with the functionality of said first software component but having incompatible data structures,

said first software component having a first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said first software component,

said first software component having a dedicated second data exchange metadata description for at least one older version of said second software component, each second data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component.

19. A distributed computer system as claimed in claim 18, wherein said computer system is a client-server-type computer system, and wherein said first software component is a server software component, and wherein said second software component is a client software component.

20. A distributed computer system as claimed in claim 18 or 19, wherein said first software component is installed in a first computer, and wherein said one or more second software components are installed in one or more second computers.

21. A client-server computer system, comprising

a server software component,

one or more client software components exchanging serialized data with said server software component,

said server software component having a first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said server software component,

said server software component having a dedicated second data exchange metadata description for at least one older version of said client software component, each second data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component.

22. A client-server computer system as claimed in claim 21, comprising a routine configured to create a serialization scheme for said server software component and said older version of said client software component from said metadata descriptions of these versions, said serialization scheme containing information needed in said server software component for the serialization of data to and the deserialization of data from said older version of said client software component.

23. A client-server computer system as claimed in claim 21 or 22, wherein said server software component comprises a serialization routine configured to transform data items into a stream of data bits to be sent to said client software component, and a deserialization routine configured to transform a stream of bits received from said client software components into data items.

24. A client-server computer system as claimed in claim 21, wherein said first software component is installed in a first computer, and wherein said one or more second software components are installed in one or more second computers.

25. A server computer for a client-server computer system, comprising means for exchanging data with client software components, said means further comprising

means for storing a first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said server software component,

means for identifying a version of a client software component,

means for selecting according to said identified version of said client component one of second data exchange metadata descriptions, said selected second data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

means for serializing data items on the basis of said first data exchange metadata description and said selected second data exchange metadata description, and sending said serialized data items to said client software component,

means for receiving serialized data items from said client software component and deserializing said data items on the basis of said first data exchange metadata description and said selected second data exchange metadata description.

26. A computer for a distributed software system comprising a first software component and at least one second software component, said computer comprising said first software component and means for exchanging data with said at least one second software component, said means further comprising

means for storing a serialization scheme for at least one pair of said new version of said first software component and an older version of said second software component, said serialization scheme containing information needed in said new version of said first software component for the serialization of data to and the deserialization of data from said previous version of said second software component, and said serialization scheme being created on the basis of a first data exchange metadata description and a second data exchange metadata description, said first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said first software component, and said second data exchange metadata description of said older version of said second software component containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

means for identifying a version of a second software component,

means for selecting one serialization scheme corresponding to said identified version of said second software component,

means for serializing data items on the basis of said selected serialization scheme,

means for receiving serialized data items from said second software component and deserializing said data items on the basis of said selected serialization scheme.

27. A server computer program comprising a program code configured to perform the following routines when run on a computer,

identifying a version of a client software component,

selecting according to said identified version of said client software component of first data exchange metadata descriptions, said selected first data exchange metadata description containing information on data structures to be used in a serialized data exchange by the identified version of said client software component,

serializing data items using data structures according to said selected first data exchange metadata description and a second data exchange metadata description containing information on data structures to be used in a serialized data exchange by said server software component,

sending said serialized data items to said client software component,

receiving serialized data items from said client software component and deserializing said data items using data structures according to said selected first data exchange metadata description and a second data exchange metadata description.

28. A server computer program as claimed in claim 27, further configured to perform the following routine:

create a serialization scheme for said server software component and said identified version of said client software component on the basis of said metadata descriptions of these versions, said serialization scheme containing information needed in said server software component for the serialization of data to and the deserialization of data from said version of said client software component.

29. A server computer program as claimed in claim 27 or 28, wherein said program code is further configured to perform the following routines:

transforming data items into a stream of data bits to be sent to said client software component,

transforming a stream of bits received from said client software components into data items.

30. A server computer program as claimed in claim 27 or 28, wherein said program code is further configured to perform the following routines:

assigning a short identifier for each instance of a data structure transferred in a data exchange,

serializing and sending a short identifier of a data structure instead of the actual data structure when such short identifier is available.

31. A server computer program comprising a program code configured to perform the following routines when run on a computer,

storing a serialization scheme for at least one pair of said new version of said first software component and an older version of said second software component, said serialization scheme containing information needed in said new version of said first software component for the serialization of data to and the deserialization of data from said respective previous version of said second software component, and said serialization scheme being created on the basis of a first data exchange metadata description and a second data exchange metadata description, said first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said first software component, and said second data exchange metadata description of said older version of said second software component containing information on data structures to be used in a serialized data exchange by the respective version of said second software component,

identifying a version of a second software component,

selecting a serialization scheme corresponding to said identified version of said second software component,

serializing data items on the basis of said selected serialization scheme,

receiving serialized data items from said second software component and deserializing said data items on the basis of said selected serialization scheme.

32. A computer program comprising program code means for performing all the steps of any one of claims 1 to 17 when the program is run on a computer.